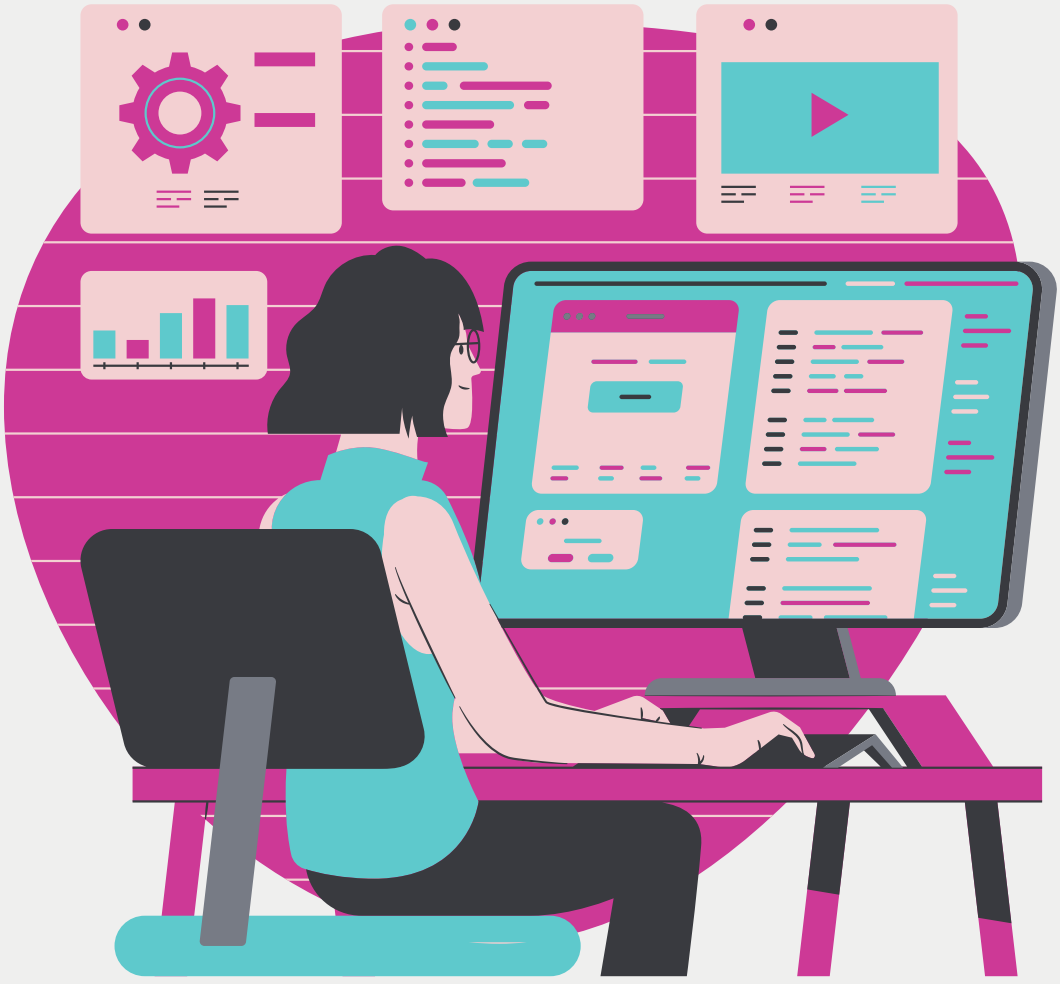
Programação Funcional



CEUB

Lucas Angelim e Maria Paula Kimura

Programação Funcional

Responsabilidades:

Pesquisa realizada por todos os integrantes.
Códigos, Wiki e formatação do Git por Lucas Angelim.
Design, material e formatação por Maria Paula Kimura.

Material:

Apresentação visual por meio de Slides;

Texto em PDF para apresentação de conteúdo;

Wiki disponibilizada no link:
https://github.com/ikimychan/Paradigma_Funcional/wiki

Introdução:

A estruturação funcional é um paradigma da programação estruturado em funções. Pode ser considerado também um estilo de estruturação de código. Embasada em princípios de matemática computacional estrutura em blocos formados por funções chaves. É otimizado para projetos e sistemas robustos e complexos.

Programação Funcional

História:

A programação funcional surge com base em um dos princípios da matemática computacional, o cálculo de lambda. Este fundamento é responsável pelo estudo das funções recursivas computáveis no contexto da teoria da computabilidade e fenômenos relacionados, como as variáveis conectadas ou substituíveis. Dessa forma, a relação estabelecida entre o cálculo-λ e a programação funcional surge por meio da representação matemática do comportamento das funções determinada na tese de Church, sendo então estabelecido o princípio do desenvolvimento do paradigma funcional no ano de 1941, em dois trabalhos do matemático Alonzo Church.

A partir da década de 1950, houve avanço significativo no estabelecimento do estilo de programação após o desenvolvimento das primeiras versões do LISP, que utilizava como base os cálculos de Church. Posteriormente, na década de 1960, surgiram também as linguagens Iswim e APL que investiram de forma intensa no paradigma estruturado em funções.

Em 1978, o cientista da computação John Backus apresentou um artigo no qual questionava o uso dos paradigmas imperativos e retratava a funcional como peça de destaque no desenvolvimento de demais linguagens funcionais de desenvolvimento.

LISP:

No fim da década de 1950, surgiu a linguagem LISP, desenvolvida e própria para a construção em funções. Até a atualidade LISP segue com popularidade e versões aprimoradas fazendo parte de seu dialeto como as variações Commom Lisp, elisp e Scheme. Portanto, ao relatar a história da programação funcional, a linguagem recebe destaque pelos seus usos ainda atuais.

Lucas Angelim e Maria Paula Kimura

Programação Funcional

Definição:

O paradigma funcional se fundamenta na utilização de funções. Sua estrutura conta com uma divisão bem definida e separada em blocos, sendo a resolução atingida por meio de criação de variáveis inseridas em funções que, em geral, buscam retornar resultados. Assim, o código base é representado de forma otimizada e organizada. A estrutura é extremamente dependente de uma base fundamentada em princípios matemáticos, principalmente exemplos como o cálculo-λ , que interpretará o problema em forma de cálculos, atingindo a solução final.

Uso de funções:

As funções que são fundamentais até para a nomeação deste paradigma se estabelecem para que a estrutura do código seja facilmente navegável e localizável, sendo cada função estabelecida e dificilmente alterada, causando instruções bem estruturadas e bem argumentadas, dispensando o uso de imperatividade no desenvolvimento do programa.

Funções Puras:

Na programação funcional existe uma função ideal, denominadas funções puras. Estas possuem resultados que são exclusivamente dependentes do que já foi inserido nos parâmetros, resultando em um código sem efeitos colaterais, pois estas funções não tem impacto externo se não retornar o valor inserido. Estas proporcionam simplicidade arquitetural para o código, evitando loops.

Funções de primeira classe:

Um dos pilares do paradigma funcional prático são as funções de primeira classe que são funções independentes. Estas buscam converter funções em dados. Desta forma, as funções criadas como de primeira classe tem valores que podem ser realocados, manipulados e retornados por outros comandos no código.

Funções de alta ordem:

As funções de alta ordem são ideais para a estruturação e desenvolvimento do código, uma vez que estas são responsáveis por receber uma função qualquer e retornar outra função, sendo que esta utiliza uma ou mais funções como argumento.

Lucas Angelim e Maria Paula Kimura

Programação Funcional

Imutabilidade:

Ao falar sobre o paradigma funcional, deve-se também considerar a filosofia inerente ao tema de não modificar dados fora das funções. Esta prática significa evitar modificar a entrada de argumentos de uma função, trazendo como principal benefício evitar efeitos colaterais, uma vez que se houver um erro, apenas influencia na função defeituosa, facilitando a localização e a eliminação de erros, principalmente, em um projeto robusto e extenso.

Vantagens:

As vantagens do uso dos paradigmas funcionais são extensas, portanto, para listar os destaques, temos que levar em consideração a confiabilidade e estabilidade proporcionada pelo código estruturado em blocos, possuindo busca simplificada por eventuais erros e bugs, permitindo com que essa seja empregada em sistemas de larga escala.

Desvantagens:

Quando há comparação entre o paradigma funcional e outros, em geral, essa possui o desempenho mais lento para processamento e para sua construção. Além da manutenção dificultada por objetos que sofrem evoluções ou valores imutáveis, sua reutilização se torna complicada e requer refatoração. O código também pode ser dificultado para iniciantes.

Programação Funcional

Atualidade:

Os paradigmas funcionais recebem destaque nos dias de hoje, sendo base para muitas outras linguagens funcionais como Haskell, ML, Clojure, Scheme, Erlang e outros. Dessa forma, sua aplicação continua consistente, apresentando destaque quando comparada aos paradigmas orientados a objeto, tendo liderança em interesse desde o ano de 2015, como apresentada na pesquisa de interesse da Google.

Casos:

Ao se tratar do emprego do paradigma funcional, podemos observá-lo em inúmeras funcionalidades e aplicações. Um exemplo claro é o estabelecimento de funções como processo da heurística no contexto de Inteligência Artificial, assim como é possível utiliza-lo como base para filtrar big data com as funções map, reduce e filter, sendo aplicável também em casos de análises estatísticas. Sendo assim, é possível concluir que os paradigmas funcionais podem ser uma linguagem comum para facilitar o entendimento e o desenvolvimento, dessa forma, um programador programará para que os demais também desenvolvam.

Lucas Angelim e Maria Paula Kimura

Programação Funcional

Referências:

[Paradigmas de programação: o que são e quais os principais?](https://blog.betrybe.com/tecnologia/paradigmas-de-programacao/)

[Can programming be liberated from the von Neumann style?: a functional style and its algebra of programs.] (https://dl.acm.org/doi/abs/10.1145/359576.359579)

[Higher Order Functions: o que são?] (https://www.alura.com.br/artigos/high-order-functions)

[An introduction to functional programming] (https://codewords.recurse.com/issues/one/an-introduction-to-functional-programming)

[What is functional programming? A practical guide] (https://www.infoworld.com/article/3613715/what-is-functional-programming-a-practical-guide.html)
https://towardsdatascience.com/why-developers-are-falling-in-love-with-functional-programming-13514df4048e

[Conception, Evolution, and Application of Functional Programming Languages] (http://www.dbnet.ece.ntua.gr/~adamo/languages/books/p359-hudak.pdf)

[Why Functional Programming Matters] (http://www.cse.chalmers.se/~rjmh/Papers/whyfp.pdf)

Programação Funcional

Referências:

[The rise of functional programming & the decline of Angular 2.0](http://blog.wolksoftware.com/the-rise-of-functional-programming-and-the-death-of-angularjs)

[Functional Programming in Data Science Projects](https://towardsdatascience.com/functional-programing-in-data-science-projects-c909c11138bb)

[Should All Scientists Choose A Functional Programming Language?](https://medium.com/chifi-media/should-all-scientists-choose-a-functional-programming-language-fa95f4b2d67a)

[The poetry of programming | Linda Liukas | TEDxCERN](https://www.youtube.com/watch?v=-jRREn6ifEQ)

[The Only Video About Functional Programming You'll Need To See](https://www.youtube.com/watch?v=miW7N2RsYpI)

[What is Functional Programming?](https://www.youtube.com/watch?v=KHojnWHemOO)

[Scrum: How to do twice as much in half the time | Jeff Sutherland | TEDxAix](https://www.youtube.com/watch?v=s4thQcgLCqk)

[Everyone Is A Software Developer | Scott Brinker | TEDxBocaRaton](https://www.youtube.com/watch?v=c2sNTAaILdA)

Referências:

[A Tutorial Introduction to the Lambda Calculus]
(https://personal.utdallas.edu/~gupta/courses/apl/lambda.pdf)

[A Brief History of Functional Programming]
(http://www.cse.psu.edu/~gxt29//historyOfFP/historyOfFP.html#Church32/33)

[Functional vs. Procedural Programming Language]
(https://web.archive.org/web/20071113175801/http://amath.colorado.edu/computing/mmm/funcproc.html)